PATENT
5760-14500
VRTS 0552

"EXPRESS MAIL" MAILING LABEL NUMBER
EL990142548US
DATE OF DEPOSIT: 12-16-03
I HEREBY CERTIFY THAT THIS PAPER OR FEE IS BEING DEPOSITED WITH THE UNITED STATES POSTAL SERVICE "EXPRESS MAIL POST OFFICE TO ADDRESSEE" SERVICE UNDER 37 C.F.R. §1.10 ON THE DATE INDICATED ABOVE AND IS ADDRESSED TO THE COMMISSIONER FOR PATENTS, P.O. BOX 1450, ALEXANDRIA, VA 22313-1450

Derrick Brown

GRAPHICAL DISPLAY FOR APPLICATION PERFORMANCE MONITORING SYSTEM

By:

Lior Porat
Shiri Kerman-Hendell
Simcha Landov

Atty. Dkt. No.: 5760-14500

B. Noël Kivlin /CLJ
Meyertons, Hood, Kivlin, Kowert & Goetzel, P.C.
P.O. Box 398
Austin, TX 78767-0398
Ph: (512) 853-8800

BACKGROUND OF THE INVENTION

Field of the Invention

[0001] This invention relates to the field of computer processing and, more particularly, to the display of computer system performance information.

Description of the Related Art

[0002] In the information technology (IT) departments of modern organizations, one of the biggest challenges is meeting the increasingly demanding service levels required by users. With more and more applications directly accessible to customers via automated interfaces such as the world wide web, "normal" business hours for many enterprises are now 24 hours a day, 7 days a week. The need for continuous availability and performance of applications has created complex, tiered IT infrastructures which often include web servers, middleware, networking, database, and storage components. These components may be from different vendors and may reside on different computing platforms. A problem with any of these components can impact the performance of applications throughout the enterprise.

[0003] The performance of key applications is a function of how well the infrastructure components work in concert with each other to deliver services. With the growing complexity of heterogeneous IT environments, however, the source of performance problems is often unclear. Consequently, application performance problems can be difficult to detect and correct. Furthermore, tracking application performance manually can be an expensive and labor-intensive task. Therefore, it is usually desirable that application performance management tasks be automated.

[0004] Automated tools for application performance management may assist in providing a consistently high level of performance and availability. These automated tools may result in lower costs per transaction while maximizing and leveraging the resources that

have already been spent on the application delivery infrastructure. Automated tools for application performance management may give finer control of applications to IT departments. Application performance management tools may enable IT departments to be proactive and fix application performance issues before the issues affect users. Historical performance data collected by these tools can be used for reports, trending analyses, and capacity planning. By correlating this collected information across application tiers, application performance management tools may provide actionable advice to help IT departments solve current and potential problems.

[0005] In a real-world environment, the performance of applications may be highly variable due to normal variations in resource usage over time. Furthermore, requirements such as user needs, usage patterns, customization requirements, system components, architectures, and platform environments may vary from business unit to business unit. These variations may also cause application performance to be highly variable. Tuning applications to work together efficiently and effectively in their unique environments can be crucial to reaching organizational goals and maintaining competitive advantages. Automated tools for application performance management can assist in these tuning operations.

SUMMARY OF THE INVENTION

[0006] Various embodiments of a system and method for providing a graphical display for an application performance monitoring system are disclosed. In one embodiment, a method may comprise tracking one or more attributes associated with each of a plurality of application tiers, displaying a plurality of objects each corresponding to a respective application tier, and altering the appearance of the corresponding object to reflect a change in the one or more attributes associated with a given application tier. In one implementation, each of the objects may include a core object and one or more indicators in proximity to the core object. Furthermore, each of the indicators may correspond to a different attribute of the associated application tier, and may change color depending on the status of the application tier. In addition, each of the one or more of objects may be connected by a directional arrow representing the data flow between the plurality of application tiers.

BRIEF DESCRIPTION OF THE DRAWINGS

[0007] Fig. 1 is a block diagram of one embodiment of a performance management system.

[0008] Fig. 2 is a block diagram of one embodiment of a computer system.

[0009] Fig. 3 is an exemplary diagram of an application tier status display according to one embodiment.

[0010] Fig. 4 is a flowchart illustrating one embodiment of a method for displaying application tier status.

[0011] While the invention is susceptible to various modifications and alternative forms, specific embodiments are shown by way of example in the drawings and are herein described in detail. It should be understood, however, that drawings and detailed description thereto are not intended to limit the invention to the particular form disclosed, but on the contrary, the invention is to cover all modifications, equivalents and alternatives falling within the spirit and scope of the present invention as defined by the appended claims.

DETAILED DESCRIPTION

[0012] A performance management system may include one or more software programs for application performance management. By continuously monitoring key components and/or applications of computer systems, the performance management system may act to detect and correct performance problems among applications and other system components in a complex computing environment. The performance management system may provide performance management in a variety of stages, such as: identification of symptoms that could indicate a performance problem, identification of sources or locations of problems, discovery of root causes of problems, recommendation of measures to address the root causes and improve performance, and verification that the measures have achieved desired goals. By defining baselines for "normal" application behavior, the performance management system may automatically detect degradation based on those established norms.

[0013] In one embodiment, the performance management system may be implemented in a variety of versions, each of which is customized for management of a particular class of target software: e.g., various products from PeopleSoft, Inc.; Oracle® database management software and related applications; web-based applications; SAP®; various products from Siebel Systems, Inc.; ClarifyCRM™; J2EE™; and other suitable targets. Furthermore, each of the versions may be implemented on one or more computing platforms (e.g., Solaris running on Sun Microsystems™ hardware, or a Microsoft Windows® OS running on Intel-based hardware). As used herein, the term "performance management system" is intended to include all of these disparate, customized software programs.

[0014] Figure 1 is an architecture diagram of a performance management system 100 in an exemplary configuration. As illustrated in Figure 1, the performance management system 100 may include components such as a measurement component 102 (including

various agent modules 104a, 106a, and 108a), a discovery component 112, a console component 120, and a performance warehouse 110. The various components of the performance management system 100 may reside on the same computer system, on different computer systems, or on an arbitrary combination of computer systems. An exemplary computer system is illustrated in Figure 2.

[0015] In one embodiment, the measurement component 102 uses agent software to capture performance metrics on servers running target applications. The measurement component 102 may provide a "breadth-wise" view of performance across multiple technology tiers (e.g., web clients, web servers, networks, application servers, database servers, storage servers, etc.). The measurement component 102 may measure, for example, end-to-end response times from the viewpoint of a user. The measurement component 102 may measure segmented response times from tier to tier and may therefore indicate the location of performance problems in a particular tier.

[0016] In one embodiment, a "base" version of the measurement component 102 may provide monitoring of a limited set of targets (e.g., TCP/IP-based applications). The functionality of the measurement component 102 may be augmented with optional agent modules that are customized to gather and analyze data for particular targets (e.g., web clients, web servers, networks, application servers, database servers, storage servers, etc.). For purposes of illustration and example, three agent modules 104a, 106a, and 108a are shown. Other combinations of agent modules may be used in other configurations.

[0017] In one embodiment, the discovery component 112 provides identification and resolution of root causes of performance degradation. By permitting a user to "drill down" through various tiers of hardware and software (e.g., individual servers), the discovery component 112 may provide a "depth-wise" view of performance within each of the tiers that a target application crosses. The discovery component 112 may further indicate steps to be taken to fix current problems or avoid future problems.

[0018] In Figure 1, each of the server blocks 104b, 106b, and 108b within the discovery component 112 are intended to represent installations of agent software on the respective servers. For example, the three database server blocks 104b represent three agent software modules associated with three respective database server installations. Likewise, the two application server blocks 106b represent two agent software modules associated with three respective application server installations, and the four storage server blocks 108b represent four agent software modules associated with four respective storage server installations. The combination of servers 104b, 106b, and 108b is provided for purposes of illustration and example and is not intended to be limiting.

[0019] In one embodiment, the console component 120 includes a "watchdog" layer that communicates key performance indicators, such as exceptions to service level agreements (SLAs), to appropriate users at appropriate times. The console component 120 may include functionality 122 for establishing SLAs and other thresholds. The console component 120 may include functionality 124 for reporting and charting. The console component 120 may include functionality 126 for providing alerts. Therefore, the console component 120 may function as a management console for user interaction with the measurement component 102 and discovery component 112.

[0020] In one embodiment, the performance warehouse 110 includes a repository of performance metrics which are accessible to the other components in the performance management system 100. For example, the historical data in the performance warehouse 110 may be used by the other components to provide short- and long-term analysis in varying degrees of detail.

[0021] The performance management system 100 of Figure 1 may be executed by one or more networked computer systems. Figure 2 is an exemplary block diagram of such a computer system 200. The computer system 200 includes a processor 210 and a memory

220 coupled together by communications bus 205. The processor 210 can be a single processor or a number of individual processors working together. The memory 220 is typically random access memory (RAM), or some other dynamic storage device, and is capable of storing instructions to be executed by the processor 210. The memory 220 may store temporary variables or other intermediate information during the execution of instructions by the processor 210. The memory 220 may store operating system (OS) software to be executed by the processor 210.

[0022] In various configurations, the computer system 200 may include devices and components such as a keyboard & mouse 250, a SCSI interface 252, a network interface 254, a graphics & display device 256, a hard disk 258, and/or a CD-ROM 260, all of which are coupled to the processor 210 by a communications bus 207. The network interface 254 may provide a communications link to one or more other computer systems via a LAN (local area network), WAN (wide area network), internet, intranet, or other appropriate networks. It will be apparent to those having ordinary skill in the art that the computer system 200 can also include numerous elements not shown in the figure, such as additional storage devices, communications devices, input devices, and output devices, as illustrated by the ellipsis.

[0023] Turning now to Fig. 3, a diagram of one embodiment of an application tier status display 300 is shown. As described above, performance management system 100 may be operable to track and analyze the performance of various functional groups of servers, or "application tiers", in a complex computing environment such as an enterprise system. For example, performance management system 100 may be operable to provide a "breadth-wise" view of performance across multiple application tiers, such as the interaction between a web server tier, an application server tier, and a database server tier. As will be described in greater detail below, application tier status display 300 may accordingly be operable to display various breadth-wise performance and status data in a

clear, simple manner, thereby allowing end users to better understand and react to application tier performance issues.

[0024] In various embodiments, application tier status display 300 may be displayed on a computer executing performance management system 100 as described above, or may be displayed on a remote computer connected to performance management system 100 via the Internet. Furthermore, application tier status display 300 may be HTML or XML content displayed on a web browser running on the above computer, or may be part of a platform-independent applet or application run by the web browser in a language such as Java or Javascript, as will be described in further detail below.

[0025] As illustrated in Fig. 3, application tier status display 300 may include a plurality of display objects 302 – 310, each corresponding to an application tier in a complex computing environment. One exemplary complex computing environment may include a web client, operable to send requests to and receive web content from a web server tier. The web server tier may in turn be operable to communicate with an application server tier, such as a Java 2 Enterprise Edition (J2EE) server cluster, which may further be operable to communicate with a SAP R/3 server tier and an Oracle database server tier. Additional tiers, as well as "cross-application" tiers that tie the various application tiers, may be possible as well. Each application tier may accordingly be represented by an associated display object 302 – 310, each of which is operable to display the status of each application tier through the use of color-coded indicators, as will be described further below.

[0026] In addition, application tier status display 300 may depict the dataflow relation of each application tier by displaying directional arrows between display objects 302 – 310. For example, the directional arrows linking web client display object 302 to web server display object 304, and web server object 304 to J2EE display object 306 may represent the flow of a service request through the various application tiers.

[0027] Each display object 302 – 310 may include a core object that identifies the associated application tier. For example, display object 306 comprises a core object with a text field reading "J2EE", indicating that display object 306 represents information about the status of the J2EE application server tier.

[0028] In one embodiment, each core object may be wholly or partially surrounded by a plurality of indicators, each representing the status of a different aspect of the associated application tier. For example, as indicated by the legend on the left side of Fig. 3, the four indicators on the top part of display objects 302 – 310 may indicate information on the current performance of the associated application tier, the application tier's performance trend over time, the application tier's current load status, and the application tier's load trend over time. Likewise, the indicators on the bottom part of a display object may represent the maintenance status of the associated application tier, the service status of the application tier, and a custom indicator.

[0029] Furthermore, each indicator in a display object may change colors to indicate information about the respective aspect of the associated application tier. As further indicated by the legend on the left side of Fig. 3, each indicator may be colored blue, yellow, or red, to indicate no alert, a near critical alert, or a critical alert, respectively. In various embodiments no alert may signify that an application tier is behaving within normal parameters, as described above, while a near critical alert signifies that an application tier is behaving abnormally, and may be close to seriously malfunctioning. Similarly, a critical alert may indicate that an application is malfunctioning and failing to provide service to client requests.

[0030] For example, as illustrated by Fig. 3, the custom, maintenance, service, and load trend indicators of J2EE application tier display object 306 are colored blue, thus indicating that there is nothing abnormal with these respective aspects of the J2EE

application server. However, the load indicator of display object 306 is colored yellow, to indicate that the load of the J2EE application tier is near critical, while the performance and performance trend indicators of display object 306 are colored red, to indicate that the current performance and the performance over time for the J2EE application tier are at critical levels.

[0031] As described above, such indications may represent a departure from the "normal" profile determined by performance monitoring system 100, and may accordingly represent, for example, an abnormally low load on the J2EE application tier represented by display object 306. Alternatively, the exemplary indicators illustrated in Fig. 3 may represent operational thresholds for the associated application tier. For example, the load indication of display object 306 may only change color when the J2EE application server tier encounters too great a load.

[0032] It is thus noted that by grouping all the indicators associated with a given application tier around a core object, and associating various status alerts with a simple color scheme, application tier status display 300 may advantageously provide a simple, easy-to-understand mechanism for communicating the status of various application tiers.

[0033] It is further noted that many of the aspects of application tier status display 300 are exemplary, and other embodiments of the display are possible. For example, a greater or lesser number of indicators may be associated with each display object, rather than the seven indicators illustrated by Fig. 3. Furthermore, it is noted that a different color scheme may be used. For example, a no alert status may be represented by green, rather than blue. Furthermore, in one embodiment a gradual color change scheme may be used to indicate a variable status aspect. For example, the load indicator of a display object may shift through dozens of color shades from blue to green to yellow to orange to red as the load on the associated application tier grows from acceptable levels (blue, green) to

critical levels (yellow, orange, red). Alternatively, in other embodiments visual cues other than color (e.g. "flashing" or animation) may be used to indicate alert status.

[0034] In other embodiments, the precise shape and/or grouping of the indicators, core object, and display object may be different. For example, rather than the double arcs shown in Fig. 3, a continuous circle of indicators or any other indicator layout scheme may be used. Alternatively, in some embodiments the display object may be a single object rather than a core object and outlying indicators, with various portions of the single display object operable to indicate the status of various aspects of the associated application tier.

[0035] It is further noted that in various embodiments, the data flow lines between display objects 302 – 310 may be bi-directional, non-directional, or not present. In addition, the color and aspect indicator legend on the left side of Fig. 3 may or may not be part of application tier status display 300, or may be toggled on and off by the user.

[0036] Fig. 4 is a flowchart illustrating one embodiment of a method for rendering the application tier status display 300 of Fig. 3. Referring collectively now to Figs. 1 – 3, in step 400 the application tiers and performance management system 100 are installed on the arbitrary combination of computer systems described above. In one embodiment, information on static parameters of the various application tiers may then be written to a data repository such as performance warehouse 110. These static parameters may include, but not be limited to, information on the number of application tiers, the data flow connections of application tiers, and metadata describing how information on the various status of aspects of the application tiers will be stored in the data repository.

[0037] In 402, the application tiers begin servicing requests, and performance management system 100 begins to monitor the performance and status of the application tiers. As described above, measurement component 102 and discovery component 112

may be operable to track the breadth-wise and depth-wise performance of each application tier, and store this information to performance warehouse 110. This information may in turn be analyzed and displayed by console component 120.

[0038] In step 404, application tier status display 300 begins to display information about the status of the various application tiers, as described above. In one embodiment application tier status display 300 may be rendered by console component 120. Alternatively, in one embodiment application tier status display 300 may be rendered by a separate system executing in parallel with performance management system 100. In such an embodiment, data may be retrieved from performance warehouse 110 through an interface between performance management system 100 and the parallel rendering system. In yet another embodiment performance management system 100 may not be used at all and application tier status display 300 may be rendered by other underlying application tier performance monitoring software.

[0039] In one embodiment, status data may be provided to application tier status display 300 via an HTML or XML document, which may in turn describe information on the placement and color of various display objects 302 – 310. In various embodiments each display object, core object, and/or indicator may be an image file such as a GIF or JPEG file, while in other embodiments display objects 302 –310 may be drawn through the use of a programming or animation language such as Java or Flash.

[0040] In 406, the tier status data in the data repository may be updated by performance management system 100. In various embodiments these updates may be periodic, continuous, or irregular and event-driven. In 408, application tier status display 300 retrieves the tier status data from the data repository. In various embodiments, this retrieval may be synchronized with data updates from performance management system 100, or may be retrieved on a separate schedule by the web browser displaying application tier status display 300. The method may then return to 402, wherein

application tier status display 300 displays the updated application tier status data, as described above. Steps 402 – 408 may thus be cycled through indefinitely, as long as performance management system 100 continues to track and display performance data.

[0041] It is noted that in one embodiment, application tier status display 300 may further be operable to provide access to more detailed information regarding various application tiers. For example, in one embodiment an indicator which indicates a critical alert for the current load of an application tier may also function as a hyperlink to a screen detailing the current status of that particular application tier. In turn, this application tier-specific screen may provide access to further information on specific aspects or components of the application tier. Accordingly, an end user may thereby be able to "drill down" from the top-level application tier status display 300 to determine specific information about an abnormal application tier status.

[0042] It is further noted that any of the embodiments described above may further include receiving, sending or storing instructions and/or data that implement the operations described above in conjunction with Figs. 1 – 4 upon a computer readable medium. Generally speaking, a computer readable medium may include storage media or memory media such as magnetic or optical media, e.g. disk or CD-ROM, volatile or non-volatile media such as RAM (e.g. SDRAM, DDR SDRAM, RDRAM, SRAM, etc.), ROM, etc. as well as transmission media or signals such as electrical, electromagnetic, or digital signals conveyed via a communication medium such as network and/or a wireless link.

[0043] Although the embodiments above have been described in considerable detail, numerous variations and modifications will become apparent to those skilled in the art once the above disclosure is fully appreciated. It is intended that the following claims be interpreted to embrace all such variations and modifications.